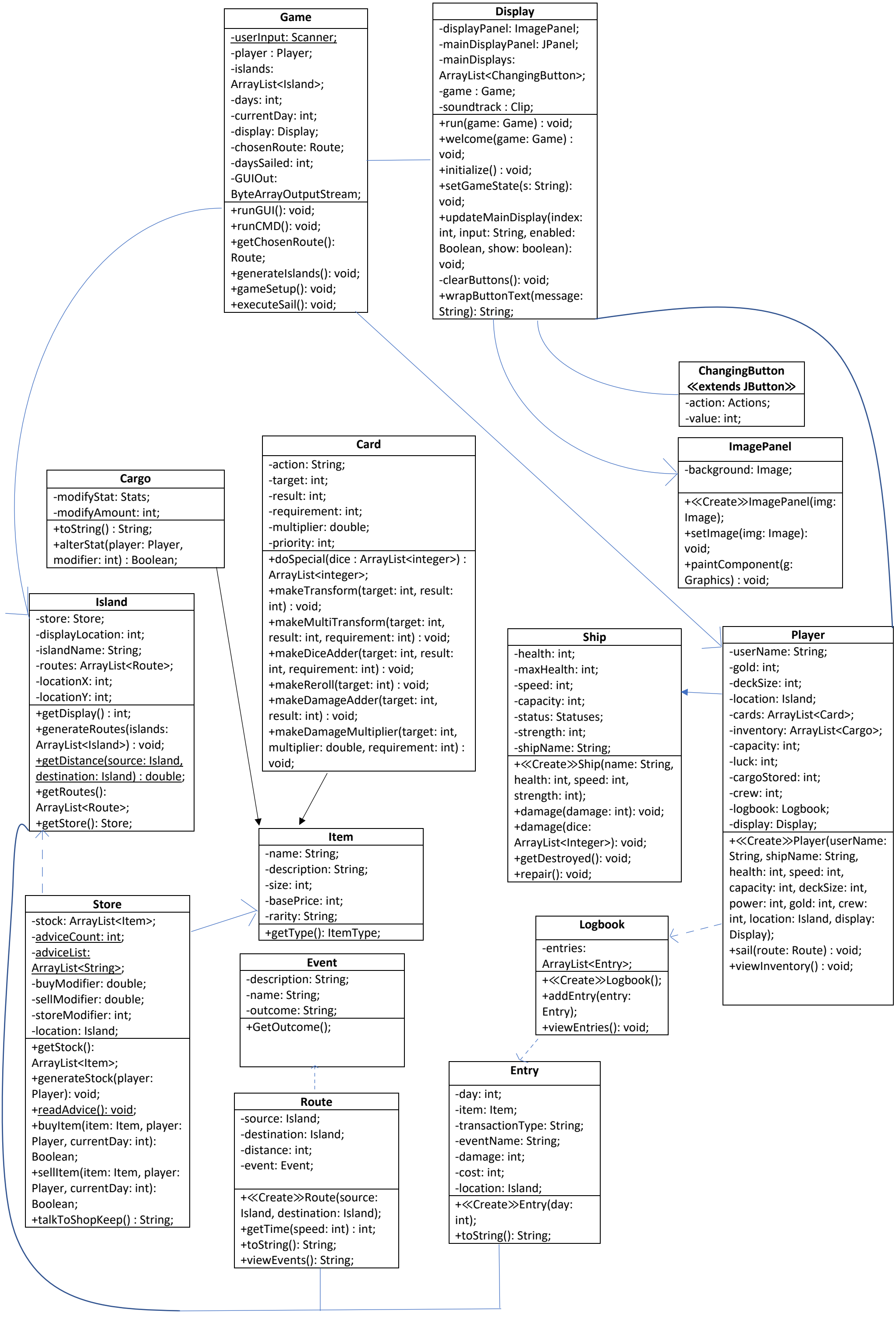
Game
-userInput: Scanner;
-player : Player;
-islands: ArrayList<Island>;
-days: int;
-currentDay: int;
-display: Display;
-chosenRoute: Route;
-daysSailed: int;
-GUIOut: ByteArrayOutputStream;
+runGUI(): void;
+runCMD(): void;
+getChosenRoute(): Route;
+generateIslands(): void;
+gameSetup(): void;
+executeSail(): void;
Display
-displayPanel: ImagePanel;
-mainDisplayPanel: JPanel;
-mainDisplays: ArrayList<ChangingButton>;
-game : Game;
-soundtrack : Clip;
+run(game: Game) : void;
+welcome(game: Game) : void;
+initialize() : void;
+setGameState(s: String): void;
+updateMainDisplay(index: int, input: String, enabled: Boolean, show: boolean): void;
-clearButtons(): void;
+wrapButtonText(message: String): String;
ChangingButton ≪extends JButton≫
-action: Actions;
-value: int;
ImagePanel
-background: Image;
+≪Create≫ImagePanel(img: Image);
+setImage(img: Image): void;
+paintComponent(g: Graphics) : void;
Cargo
-modifyStat: Stats;
-modifyAmount: int;
+toString() : String;
+alterStat(player: Player, modifier: int) : Boolean;
Card
-action: String;
-target: int;
-result: int;
-requirement: int;
-multiplier: double;
-priority: int;
+doSpecial(dice : ArrayList<integer>) : ArrayList<integer>;
+makeTransform(target: int, result: int) : void;
+makeMultiTransform(target: int, result: int, requirement: int) : void;
+makeDiceAdder(target: int, result: int, requirement: int) : void;
+makeReroll(target: int) : void;
+makeDamageAdder(target: int, result: int) : void;
+makeDamageMultiplier(target: int, multiplier: double, requirement: int) : void;
Island
-store: Store;
-displayLocation: int;
-islandName: String;
-routes: ArrayList<Route>;
-locationX: int;
-locationY: int;
+getDisplay() : int;
+generateRoutes(islands: ArrayList<Island>) : void;
+getDistance(source: Island, destination: Island) : double;
+getRoutes(): ArrayList<Route>;
+getStore(): Store;
Ship
-health: int;
-maxHealth: int;
-speed: int;
-capacity: int;
-status: Statuses;
-strength: int;
-shipName: String;
+≪Create≫Ship(name: String, health: int, speed: int, strength: int);
+damage(damage: int): void;
+damage(dice: ArrayList<Integer>): void;
+getDestroyed(): void;
+repair(): void;
Player
-userName: String;
-gold: int;
-deckSize: int;
-location: Island;
-cards: ArrayList<Card>;
-inventory: ArrayList<Cargo>;
-capacity: int;
-luck: int;
-cargoStored: int;
-crew: int;
-logbook: Logbook;
-display: Display;
+≪Create≫Player(userName: String, shipName: String, health: int, speed: int, capacity: int, deckSize: int, power: int, gold: int, crew: int, location: Island, display: Display);
+sail(route: Route) : void;
+viewInventory() : void;
Item
-name: String;
-description: String;
-size: int;
-basePrice: int;
-rarity: String;
+getType(): ItemType;
Store
-stock: ArrayList<Item>;
-adviceCount: int;
-adviceList: ArrayList<String>;
-buyModifier: double;
-sellModifier: double;
-storeModifier: int;
-location: Island;
+getStock(): ArrayList<Item>;
+generateStock(player: Player): void;
+readAdvice(): void;
+buyItem(item: Item, player: Player, currentDay: int): Boolean;
+sellItem(item: Item, player: Player, currentDay: int): Boolean;
+talkToShopKeep() : String;
Event
-description: String;
-name: String;
-outcome: String;
+GetOutcome();
Logbook
-entries: ArrayList<Entry>;
+≪Create≫Logbook();
+addEntry(entry: Entry);
+viewEntries(): void;
Route
-source: Island;
-destination: Island;
-distance: int;
-event: Event;
+≪Create≫Route(source: Island, destination: Island);
+getTime(speed: int) : int;
+toString(): String;
+viewEvents(): String;
Entry
-day: int;
-item: Item;
-transactionType: String;
-eventName: String;
-damage: int;
-cost: int;
-location: Island;
+≪Create≫Entry(day: int);
+toString(): String;